멀티 레이어 포인터 네트워크를 이용한 한국어 의존 구문 분석

박천음*O, 황현선*, 이창기*, 김현기**
강원대학교*, 한국전자통신연구원**
{parkce, hhs4322, leeck}@kangwon.ac.kr, hkk@etri.re.kr

Korean Dependency Parsing with Multi-layer Pointer Networks

Cheoneum Park*O, Hyunsun Hwang*, Changki Lee*, Hyunki Kim**
Kangwon National University*, Electronics and Telecommunications Research Institute**

요 약

딥 러닝 모델은 여러 히든 레이어로 구성되며, 히든 레이어의 깊이가 깊어질수록 레이어의 벡터는 높은 수준으로 추상화된다. 본 논문에서는 Encoder RNN의 레이어를 여러 층 쌓은 멀티 레이어 포인터 네트워크를 제안하고, 멀티 태스크 학습 기반인 멀티 레이어 포인터 네트워크를 이용한 한국어 의존 구문 분석 모델을 제안한다. 멀티 태스크 학습 모델은 어절 간의 의존 관계와 의존 레이블 정보를 동시에 구하여 의존 구문 분석을 수행한다. 실험 결과, 본 논문에서 제안한 모델이 기존 한국어 의존 구문 분석 연구들 보다 좋은 UAS 92.16%, LAS 89.88%의 성능을 보였다.

주제어: 의존 구문 분석, 멀티 레이어 포인터 네트워크, Multi-layer Pointer Networks, 딥 러닝

1. 서론

구문 분석은 문장성분 사이의 관계를 분석하고 문장의 구조적, 의미적 중의성을 해결하는 자연어처리 문제이며, 의존 구문 분석(Dependency parsing)과 구구조 구문 분석(Phrase structure parsing) 등이 있다. 의존 구문 분석은 문장구조를 중심어(head)와 수식어(modifier)로 구성된 의존 관계로 표현하는 방법이며[1], 어순이 자유롭고 문장성분의 생략이 빈번한 한국어와 같은 언어에서 주로 연구되었다. 최근 딥 러닝 기반 의존 구문 분석은 RNN (Recurrent neural network)를 이용한 전이 기반 방법[2-4]과 그래프 기반 방법[5-9]이 활발하게 연구되고 있다. 의존 구문 분석은 의미분석(의미역 결정 등)과 담화분석(상호참조해결), 질의응답 등에 응용될 수 있다.

포인터 네트워크(Pointer networks)[10]는 어텐션 메커니즘(Attention mechanism)[11]을 이용하여 입력열에 대응되는 위치를 출력 결과로 하는 RNN의 확장된 모델이며, 입력된 문장에서 특정 단어를 가리켜 찾는 문제에 적합하다. 어텐션 메커니즘은 주어진 입력 열 중에서 출력 결과에 영향을 미치는 위치를 더욱 집중하여 계산하는 어텐션 가중치(attention weight)를 학습하는 방법이다.

본 논문에서는 인코딩 단계에서 인코더(Encoder RNN)를 여러 층으로 쌓아 더 높은 수준의 추상화를 시도하고, 디코딩 단계에서 멀티 태스크 학습 기반 포인터 네트워크를 이용하여 각 어절의 중심어를 찾고 의존 구문 분석의 레이블(label) 정보를 출력하는 멀티 레이어 포인터 네트워크(Multi-layer Pointer Networks)를 이용한 한국어 의존 구문 분석을 제안한다.

2. 관련 연구

최근 딥 러닝을 이용한 전이 기반과 그래프 기반 의존 구문 분석이 활발히 연구되고 있다. 전이 기반 의존 구문 분석은 입력(버퍼)과 스택으로부터 구문 분석 상태 표현을 얻고, 신경망을 이용하여 다음 전이 액션을 결정하는 방법이다[2-4]. 그래프 기반 방법은 입력 단어들에 대한 의존 관계의 점수(score)를 딥 러닝 모델로 계산하여 의존 구문 분석을 수행한다[5-9].

한국어에서 딥러닝을 이용한 의존 구문 분석은 [12]의 FFNN (Feed-forward Neural Network)를 이용한 의존 구문 분석을 시작으로, RNN을 이용하여 전이 기반 의존 구문 분석을 수행하는 [2-4]와 그래프 기반 의존 구문 분석을 수행하는 [5-9]가 있다. 그 외로 네트워크 출력 결과만으로 의존 관계를 파악할 수 있는 포인터 네트워크를 이용한 의존 구문 분석[13]이 있다. 포인터 네트워크를 이용한 방법은 멀티 태스크 학습 기반으로 중심어의 위치와 레이블 정보를 동시에 학습할 수 있다.

본 논문에서 제안하는 멀티 레이어 포인터 네트워크를 이용한 한국어 의존 구문 분석은 기존 포인터 네트워크를 이용한 의존 구문 분석 모델의 인코더 부분을 여러 층으로 쌓아 인코딩을 수행하여 인코딩 레벨에서 각 단어들 간의 관계표현을 더욱 추상화하여 학습 및 예측을 수행하는 확장된 방법이다.

3. 멀티 레이어 포인터 네트워크

포인터 네트워크는 RNN encoder-decoder에서 확장된 모델로서 어텐션 메커니즘을 기반으로 한다. 인코더는 RNN을 이용하여 입력 열에 대한 hidden state를 인코딩하여 인코딩 벡터로 만든다. 디코더(Decoder RNN)는 현

재까지 생성된 디코더의 hidden state와 인코딩 벡터를 어텐션 함수로 계산하여 입력열에 대한 얼라인먼트 점수(alignment score)를 구하고, 가장 높은 점수를 가지는 위치를 결과로 출력한다. 멀티 레이어 포인터 네트워크는 인코더 단계에서 레이어를 여러 층 쌓아 구성한 포인터 네트워크의 확장된 모델이다. 인코더에서 인코딩을 수행할 때 레이어를 여러 층 쌓으면 각 hidden state의 표현들이 더욱 추상화된다.

본 논문에서는 인코더로 bidirectional Gated Recurrent Unit(BiGRU)[14]을 사용하며, 각 히든 레이어의 BiGRU 수식은 아래와 같다.

$$e_s = W_e x_s$$
$$h_s^1 = BiGRU(h_{s-1}^1, e_s)$$
$$h_s^2 = BiGRU(h_{s-1}^2, h_s^1)$$
$$h_s^3 = BiGRU(h_{s-1}^3, h_s^2)$$

x_s는 입력열의 s 번째 단어이고, e_s는 x_s에 단어표현(word embedding)을 적용한 것이다. h_s^1은 BiGRU로 인코딩한 결과로써 GRU의 forward pass와 backward pass의 hidden state를 연결(concatenate)한 것이며, 다음 히든 레이어 h_s^2의 입력으로 주어진다. h_s^2는 h_s^1을 입력으로 받아 BiGRU로 인코딩을 수행한 결과이며, h_s^3도 h_s^2를 입력으로 받아 BiGRU로 인코딩을 수행한 결과이다. 이와 같이 레이어를 여러 층 쌓게 되면 주어진 입력열의 표현을 더욱 추상화된다.

디코더에서는 입력열의 특정 위치($yi_t \in Y_{input}$)를 입력받아 yi_t의 중심어 위치($y_t \in Y_{output}$)와 의존 구문 레이블($z_t \in Z_{output}$)을 예측하며 다음과 같이 정의된다.

$$h_t = GRU(h_{yi_t}^l, h_{t-1}),\ l \in \{1,2,3\}$$
$$a_t^s = \frac{\exp(score_a(h_t, h_s^l))}{\sum_{s'} \exp(score_a(h_t, h_{s'}^l))}$$
$$y_t = argmax_s(a_t^s)$$

디코더에서는 forward GRU를 사용하며, 디코더의 각 스텝의 hidden state를 h_t로 나타낸다. h_t는 인코더의 hidden state $h_{yi_t}^l$과 디코더의 이전 hidden state를 입력으로 받아 GRU를 거쳐 계산한다. 인코더의 hidden state h_s^l에서 인덱스 l은 멀티 레이어의 수를 의미한다. a_t^s는 yi_t의 중심어 위치 확률로 $score_a$함수의 결과 벡터를 정규화한 값(attention weight)이다. a_t^s에서 가장 높은 확률을 갖는 위치가 yi_t의 중심어 위치인 출력 결과 y_t가 된다.

$$c_t^D = h_{y_t}^l$$
$$score_a(h_t, h_s^l) = \begin{cases} v_t^T \tanh(W_a[h_t; h_s^l]), & concat \\ v_t^T \tanh(W_a[h_t; h_{yi_t}^l; h_s^l]), & concat2 \\ v_t^T \tanh(W_a[h_t; c_{t-1}^D; h_s^l]), & concat3 \\ v_t^T \tanh(W_a[h_t; h_{yi_t}^l; c_{t-1}^D; h_s^l]), & concat4 \end{cases}$$

$score_a$함수는 얼라인먼트 점수를 계산하며, $concat$, $concat2$와 $concat3$, $concat4$로 나뉜다. 먼저 $concat$은 h_t와 h_s^l을 연결한(concat) 후 가중치 행렬을 곱하여 점수를 계산하며, $concat2$는 h_t와 $h_{yi_t}^l$, h_s^l을 이용하여 점수를 계산한다. $concat3$는 h_t와 c_{t-1}^D, h_s^l을 이용하여 점수를 계산하고, $concat4$는 h_t와 $h_{yi_t}^l$, c_{t-1}^D, h_s^l 모두를 이용하여 점수를 계산한다. 여기서 c_t^D는 a_t에서 가장 높은 얼라인먼트 점수를 가지는 y_t를 이용(hard attention)하여 구한 문맥 벡터이다.

$$score_z(h_t, c_t^D) = \begin{cases} u_t^T ReLU(W_z[c_t^D; h_t]), & concat_z \\ u_t^T ReLU(W_z[c_t^D; h_t; h_{yi_t}^l]), & concat2_z \end{cases}$$
$$z_t = argmax(softmax(score_z(h_t, h_{y_t}^l))$$

z_t는 의존 관계 레이블의 출력 결과로, h_t와 c_t^D를 입력으로 한 $score_z$함수를 이용하여 구한다. $score_z$함수는 $concat_z$와 $concat_z2$ 방법으로 나뉜다. $concat_z$는 $score_a$의 $concat$을 기반으로 하며, c_t^D와 h_t를 이용하여 점수를 계산한다. $concat_z2$ 방법은 $score_a$의 $concat2$와 $concat3$, $concat4$를 기반으로 하며, c_t^D와 h_t, $h_{yi_t}^l$을 이용하여 점수를 계산한다.

4. 멀티 레이어 포인터 네트워크를 이용한 한국어 의존 구문 분석

본 논문에서 제안한 멀티 레이어 포인터 네트워크를 이용한 의존 구문 분석은 멀티 태스크 학습 방법을 이용하며, 디코더의 입력으로 주어진 임의의 어절(형태소 단위로 나눠 입력)에 대하여 의존 관계를 갖는 중심어의 위치를 학습하고, 이에 해당하는 레이블 정보도 함께 학습한다. 멀티 레이어 포인터 네트워크는 인코더를 여러 층 쌓아 일반 포인터 네트워크보다 더 높은 추상화를 시도하는 모델이다. 이에 따른 멀티 레이어 포인터 네트워크의 모델 구조는 [그림 1]과 같다.

[그림 1]은 인코더와 디코더로 구성되며, 인코더는 입력열(Input layer)과 임베딩 레이어(Projection layer), l개($l \in \{1,2,3\}$)의 히든 레이어 (Hidden layer)로 구성된다. 입력열은 $X = \{A, B, C, D, </s>\}$와 같으며, 각 알파벳은 입력되는 형태소를 의미하고, </s>는 문장의 끝을 알리는 종료 기호이다. 이때 인코더의 입력인 형태소들은 문장의 어절들을 형태소 단위로 바꿔 사용하고, 어절 정보를 표현하기 위하여 어절 사이에 띄어쓰기 심볼(<sp>)을 추가한다. 인코더의 입력열 X는 임베딩 레이어에서 단어 표현(word embedding)이 적용되어 히든 레이어로 보내진다. 첫 번째 히든 레이어에서 BiGRU를 수행하여 인코딩 벡터 h^1을 만들고, 그 다음 두 번째 히든 레이어에서 앞서 생성한 h^1을 입력으로 인코딩을 수행한다. 두 번째 히든 레이어의 인코딩 결과 h^2는 세 번째 히든 레이어의 입력으로 주어 인코딩을 수행하여 h^3를 만들며, 각 레이어 마다 드랍아웃(dropout)을 적용하여 과적합

(over fitting)을 방지한다.

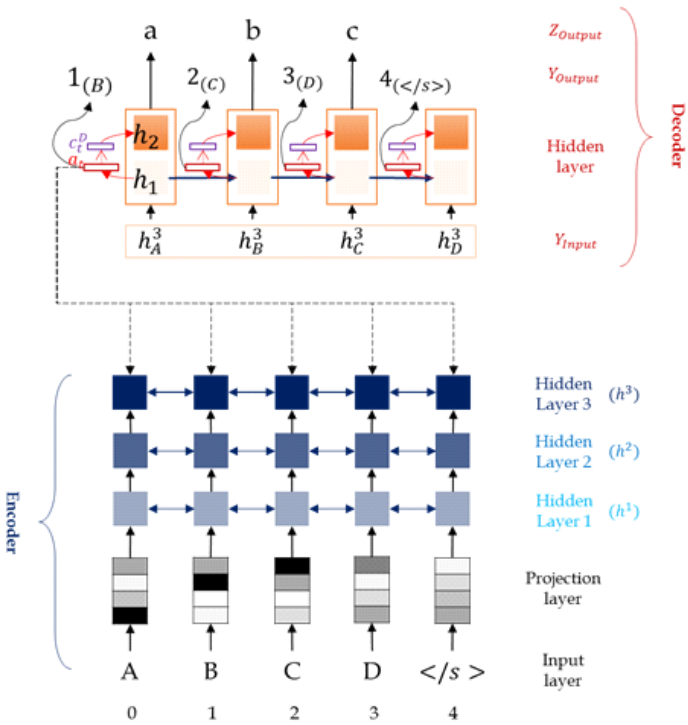

그림 1. 멀티 레이어 포인터 네트워크를 이용한 의존 구문 분석 모델

디코더는 입력열(Y_{input})과 히든 레이어, 출력 결과(Y_{output}, Z_{output})로 구성되며, Y_{input}={$0_{(A)}$, $1_{(B)}$, $2_{(C)}$, $3_{(D)}$}와 같이 주어진 입력열에 대응되는 출력열들(의존 관계를 나타내는 Y_{output}={$1_{(B)}$,$2_{(C)}$,$3_{(D)}$, $4_{(</s>)}$}, 의존 관계 레이블 정보를 출력하는 Z_{output}={a, b, c, d})을 생성한다. 디코더의 입력 Y_{input}은 각 어절의 위치 정보를 형태소 레벨의 인덱스로 표현한 것이며, 출력 Y_{output}은 각 어절의 중심어 위치를 형태소 레벨의 인덱스로 표현한 것이다. 이와 같이 입력과 출력의 단위는 형태소이기 때문에 어절의 대표 형태소 위치를 정의하여 학습데이터를 만들어 사용하며, [13]에서 가장 좋은 성능을 보인 *Dp0*를 입력기준으로 한다.

5. 실험

본 논문에서 제안한 멀티 레이어 포인터 네트워크를 이용한 한국어 의존 구문 분석의 실험 데이터는 의존 구조로 변환된 세종 데이터셋[13]을 사용하였으며, 데이터셋은 총 59,659 문장이다. 학습에 사용한 문장은 전체 문장 중 90%인 53,842 문장이고, 나머지 10%인 5,817 문장을 평가에 사용하였다. 입력 형태소에 대한 단어표현은 10만 단어에 대한 2년치 뉴스기사를 Neural Network Language Model (NNLM)으로 학습한 것을 사용하였다. 의존 구문 분석 결과에 대한 평가 척도는 Unlabeled Attachment Score (UAS), Labeled Attachment Score (LAS)를 사용하였다.

인코더와 디코더, 어텐션 레이어의 활성함수는 모두 tanh를 사용하였고, 의존 관계 레이블 Z_{output}의 활성함수는 relu를 사용하였다. 임베딩 레이어에서 적용되는 단어 표현은 형태소 단위 100차원을 사용하였고, 히든 레이어의 차원 수와 드랍아웃은 [14]에서 가장 좋은 성능을 보인 차원 수 600과, 드랍아웃 0.2로 설정하였으다. 학습은 모멘텀(momentum)을 이용하였고, 학습율(learning rate) 0.1을 시작으로 성능 개선이 없으면 3 에포크(epoch)마다 50%씩 감소시켰다.

표 1. 한국어 의존 구문 분석 성능 비교 (자동 분석 형태소 결과 이용)

의존 구문 분석	UAS	LAS
임수종[15]: 세종코퍼스	88.15	-
이창기[12] with MI	90.37	88.17
박천음[13]: Pointer Networks	91.79	89.48
나승훈[9]: Deep biaffine	91.78	89.76
2-layer stack concat	**92.16**	**89.88**
2-layer stack concat2	92.11	89.82
2-layer stack concat3	92.12	89.85
2-layer stack concat4	92.08	89.78
3-layer stack concat	92.11	89.70
3-layer stack concat2	92.13	89.79
3-layer stack concat3	91.98	89.63

[표 1]은 본 논문에서 제안한 멀티 레이어 포인터 네트워크를 이용한 한국어 의존 구문 분석 성능을 기존의 연구들과 비교한 표이다. 사용한 데이터는 모두 세종코퍼스로 같으며 자동 형태소분석 결과를 사용하였다. 실험 결과, 본 논문에서 제안한 멀티 레이어 포인터 네트워크를 이용하여 의존 구문 분석을 수행하는 방법이 일반 포인터 네트워크를 사용한 박천음[14]과 Deep biaffine 모델을 사용한 나승훈[9] 연구에 비하여 전반적으로 높은 성능을 보였으며, 2-layer stack cancat 모델이 UAS 92.16%, LAS 89.88%로 박천음[14] 연구에 비하여 UAS 0.37%, LAS 0.4%, 나승훈[9] 연구에 비하여 UAS 0.38%, LAS 0.12%의 성능 향상을 보였다. 그러나 3-layer stack을 하였을 때 오히려 성능이 떨어지는 경향을 보였다. 이는 stack을 쌓는 것이 항상 좋은 성능을 보이지 않으며 과적합 되기 쉬움을 보여준다.

[그림 2, 3]은 가장 좋은 성능을 보인 2-layer stack cancat 모델에 대한 의존 관계 포인팅 얼라인먼트 스코어와 의존 레이블 얼라인먼트 스코어에 대한 결과를 보인다. 인코더의 입력은 "덕분/NNG[0] 에/JKB[1] <SP>[2] 나/NP[3] 는/JX[4] <SP>[5] 수석/NNG[6] 으로/JKB[7] <SP>[8] 졸업/NNG[9] 하/XSV[10] 는/ETM[11] <SP>[12] 기쁨/NNG[13] 을/JKO[14] <SP>[15] 가지/VV[16] 었/EP[17] 다/EF[18] ./SF[19] </S>[20]"과 같다. 디코더는 입력 기준 *Dp0*를 따르며, 디코더의 입력은 "덕분/NNG, 나/NP, 수석/NNG, 졸업/NNG, 기쁨/NNG, 가지/VV"와 같고, 디코더의 정답은 y_{output}=[16, 16, 9, 13, 16, 20], z_{output}=[DP, NP_OBJ, DP, NP, VP_MOD, NP_AJT]와 같다. 디코더의 출력은 [그림 2]와

같이 $\hat{y}_{output}$=[16, 16, 9, 13, 16, 20], [그림 3]과 같이 $\hat{z}_{output}$=[DP, NP_OBJ, DP, NP, VP_MOD, NP_AJT]의 결과를 보였다.

6. 결론

본 논문에서는 한국어 의존 구문 분석을 수행하기 위하여 인코딩 단계에서 더욱 높은 추상화를 시도하기 위한 멀티 레이어 포인터 네트워크 모델을 이용하였다. 실험 결과, 본 논문에서 제안한 방법이 2-layer stack과 concat 얼라인먼트 스코어 방법, 단어 표현 100차원, 히든 레이어 600 차원, 드랍아웃 0.2와 같은 하이퍼 파라미터일 때 UAS 92.16%, LAS 89.88%로 한국어 의존 구문 분석에 관한 선행연구들 보다 더욱 높은 성능을 보였다.

향후 연구로는 의존 구문 분석을 수행하는 포인터 네트워크에 음절과 어절 표현을 추가로 적용하여 인코더에서의 표현력을 더욱 높인 네트워크 구조를 모델링 할 예정이다.

감사의 글

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 정보통신·방송 연구개발 사업의 일환으로 하였음. [2013-0-00131, (엑소브레인-1세부) 휴먼 지식증강 서비스를 위한 지능진화형 WiseQA 플랫폼 기술 개발]

참고문헌

[1] D. Hays. Dependency theory: a formalism and some observations. Language, pp. 511-525, 1964.

[2] M. Ballesteros, C. Dyer, N. A. Smith. Improved Transition-Based Parsing by Modeling Characters instead of Words with LSTMs, *EMNLP 2015*, 2015.

[3] L. Kong, C. Alberti, D. Andor, I. Bogatyy, D. Weiss. DRAGNN: A Transition-based Framework for Dynamically Connected Neural Networks, https://arxiv.org/abs/1703.04474

[4] 나승훈, 김강길, 김영길. Stack LSTM을 이용한 전이 기반 한국어 의존 파싱, *KCC 2016*, 2016.

[5] 나승훈, 이건일, 신종훈, 김강일. 순환 컨트롤러를 이용한 Stack LSTM기반 한국어 의존 파싱, *KCC 2016*, 2016.

[6] E. Kiperwasser and Y. Goldberg. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL*, 2016.

[7] T. Dozat, C. D. Manning. Deep Biaffine Attention for Neural Dependency Parsing, *ICLR 2017*, 2017.

[8] X. Ma and E. Hovy. Neural Probabilistic Model for Non-projective MST Parsing, https://arxiv.org/abs/1701.00874

[9] 나승훈, 이건일, 신종훈, 김강일. Deep Biaffine Attention을 이용한 한국어 의존 파싱, *KCC 2017*, 2017.

[10] O. Vinyals, M. Fortunato and N. Jaitly. Pointer Networks. *Advances in Neural Information Processing Systems*, pp. 2674-2682, 2015.

[11] D. Bahdanau, et al. Neural machine translation by jointly learning to align and translate. *Proc. of ICLR' 15*, arXiv:1409.0473, 2015.

[12] 이창기, 김준석, 김정희. 딥 러닝을 이용한 한국어 의존 구문 분석. 제26회 한글 및 한국어 정보처리 학술대회, pp. 87-91, 2014.

[13] 박천음, 이창기. 포인터 네트워크를 이용한 한국어 의존 구문 분석. *정보과학회논문지 44.8*, pp. 822-831, 2017.

[14] K. Cho, et al. Learning phrase representation using RNN encoder-decoder for statistical machine translation. *Proc. of EMNLP' 14*, 2014.

[15] 임수종, 김영태, 나동열. 자질 가중치의 기계학습에 기반한 한국어 의존파싱. 정보과학회논문지, 소프트웨어 및 응용 제38권 제4호, 2011.

의존관계 어텐션	덕분/NNG	나/NP	수석/NNG	졸업/NNG	기쁨/NNG	가지/VV
덕분/NNG [0]						
에/JKB [1]						
<SP> [2]						
나/NP [3]						
는/JX [4]						
<SP> [5]						
수석/NNG [6]						
으로/JKB [7]						
<SP> [8]						
졸업/NNG [9]			0.9966			
하/XSV [10]						
는/ETM [11]						
<SP> [12]						
기쁨/NNG [13]				0.9997		
을/JKO [14]						
<SP> [15]						
가지/VV [16]	1.0000	1.0000			1.0000	
었/EP [17]						
다/EF [18]						
./SF [19]						
</S> [20]						1.0000

그림 2. 한국어 의존 구문 분석 결과의 포인팅 얼라인먼트 점수

레이블 어텐션	덕분/NNG	나/NP	수석/NNG	졸업/NNG	기쁨/NNG	가지/VV
NP_CNJ						
NP_SBJ						
VP						
NP_OBJ		0.9994				
NP_AJT						0.9999
VP_MOD					1.0000	
DP	0.9998		0.9999			
NP				0.9999		
AP						
VNP						
NP_MOD						
VP_SBJ						
VP_OBJ						
VNP_MOD						
VP_CMP						
NP_CMP						
VNP_CMP						
VNP_OBJ						
VNP_AJT						
AP_AJT						
VP_AJT						
IP						
AP_MOD						
NP_INT						
VNP_SBJ						
L						
R						
X_CMP						
VP_CNJ						
VNP_CNJ						
X_CMP						
X_MOD						
X_CNJ						
X_AJT						
X_SBJ						
X_OBJ						
<S>						
</S>						

그림 3. 한국어 의존 구문 분석 결과의 레이블 분류에 대한 얼라인먼트 점수